Research online Publishing
www.ronpub.com

Open Access

Open Journal of Web Technologies (OJWT)
Volume 2, Issue 1, 2015
http://www.ronpub.com/ojwt
ISSN 2199-188X

Detecting Vital Documents in Massive Data Streams

Shun Kawahara [A], Kazuhiro Seki [B], Kuniaki Uehara [A]

[A] Graduate School of System Informatics, Kobe University, 1-1 Rokkodai, Nada, Kobe 657-8501, Japan, kawahara@ai.cs.kobe-u.ac.jp, uehara@kobe-u.ac.jp

[B] Faculty of Intelligence and Informatics, Konan University, 8-9-1 Okamoto, Higashinada, Kobe 658-8501, Japan, seki@konan-u.ac.jp

Abstract

Existing knowledge bases, including Wikipedia, are typically written and maintained by a group of voluntary editors. Meanwhile, numerous web documents are being published partly due to the popularization of online news and social media. Some of the web documents, called "vital documents", contain novel information that should be taken into account in updating articles of the knowledge bases. However, it is practically impossible for the editors to manually monitor all the relevant web documents. Consequently, there is a considerable time lag between an edit to knowledge base and the publication dates of such vital documents. This paper proposes a realtime detection framework of web documents containing novel information flowing in massive document streams. The framework consists of two-step filter using statistical language models. Further, the framework is implemented on the distributed and fault-tolerant realtime computation system, Apache Storm, in order to process the large number of web documents. On a publicly available web document data set, the TREC KBA Stream Corpus, the validity of the proposed framework is demonstrated in terms of the detection performance and processing time.

Type of Paper and Keywords

Regular research paper: *social media, knowledge bases, web documents, vital documents, documment streams, novel information, realtime processing*

1 Introduction

Nowadays, large knowledge bases, such as Wikipedia, are widely used as a quick reference tool to find all kinds of information in our daily lives. Here, knowledge bases are defined as collaborative online encyclopedias edited and maintained by their users. Such knowldege bases can be utilized to improve the performance of various information processing tasks, such as query expansion [9, 24], entity linking [18], question answering [6] and entity retrieval [3]. For these applications, maintaining the quality of the knowledge bases in a timely manner is crucial. Knowledge bases should be updated when information about the current state, actions, or situation of the topic of any existing article becomes available. We define such timely information as *novel information* and an article's topic as *entity*. Entity can be a person, facility, organization, or concept, such as "Barack Obama", "White House", and "Democratic Party".

Knowledge bases typically contain a large number of articles. For example, the English version of Wikipedia has over 4.5 million articles and they are maintained by small workforces of humans, i.e., about 1,300 editors[1]. If those articles are evenly split among the editors, an

[1] http://en.wikipedia.org/wiki/List_of_Wikipedias

editor would be responsible for maintaining about 3,500 articles. In the meantime, the amount of web documents continues to grow due to the exploding popularity of social networking service (SNS), such as Twitter and Facebook, across the world. Currently, editors manually monitor relevant document streams and edit articles when they happen to notice novel information. Consequently, there is a considerable time lag between the date of an edit to knowledge base and the publication date of vital documents. It is reported that the length of the time lag often become a year [11].

In this paper, we propose a realtime detection framework of such documents containing novel information flowing in massive document streams. The Text REtrieval Conference (TREC) Knowledge Base Acceleration (KBA) track [10, 12] targeted this particular problem, referred to as the "vital filtering" task. Here, vital documents are those containing novel information which would motivate an update to the entity's article of knowledge base at the time it is published. On the other hand, useful documents are those containing information related to a target entity but not contain novel information, e.g., background biography, primary or secondary source. For instance, the document "Today, Obama announced a new policy." is a vital document because this document describes timely information about the current actions of Obama. On the other hand, the document "Obama was inaugurated as president on January 20, 2009." is an useful document because this document describes an event from the past. Vital and useful documents are collectively called *relevant* documents.

Participants of the track developed a variety of systems. However, they are generally suffered from the following issues: (1) no consideration of poor training data, (2) poor performance of detection of web documents containing novel information in relevant documents, and (3) no consideration as to how to process massive document streams in realtime. To deal with these issues, we take advantage of a pseudo relevance feedback model for non-relevant documents and use two distinct statistical language models to represent documents containing novel information. Furthermore, our proposed framework is implemented on the distributed and fault-tolerant realtime computation system, Apache Storm[2], in order to process massive document streams in realtime. The present work is based on our previous work [13] and extends it by combining the two language models.

The remainder of this paper is structured as follows: Section 2 reviews representative approaches devised for the TREC KBA vital filtering task. Section 3 briefly introduces Apache Storm and its components and describes our proposed framework. Section 4 evaluates our framework by reporting the results of empirical experiments, and Section 5 concludes this paper with a brief summary and possible future directions.

[2]http://storm.apache.org/

2 Related Work

In recent years, much attention has been paid to filtering massive web documents and detecting information related to the topic. The TREC KBA vital filtering task has investigated the challenge of detecting relevant documents about a specific entity (e.g., a person, an organization, and a facility) since 2012. In 2013, the vital filtering task required participants to distinguish between relevant documents containing novel information and those are relevant but not containing novel information. Various approaches to detecting novel information in text streams have been thus developed for the TREC KBA vital filtering task in 2013. The proposed approaches fall into two categories.

The first category of approaches tackled the task as a ranking problem [8, 15]. Dietz and Dalton [8] proposed a feature expansion technique using topic information related to the target entity. Liu et al. [15] ranked documents that match the entity by leveraging the number of occurrences and weights of related topics collected from the Wikipedia page of the topic.

The second category tackled the task as a classification problem [1, 2, 4, 22] by classifying input documents as vital or useful. Abbes et al. [1] employed such a classifier with a number of features, including where keywords related to the target entity occur in a document, whether the document title mentions the keywords, etc. Balog et al. [2] proposed two multi-step classification approaches that use four types of features: (1) document features such as document length and document source; (2) topic features such as the number of related topics; (3) document-topic features such as the number of occurrences of the topic in the document; and (4) temporal features such as average hourly Wikipedia page views. Bellogín et al. [4] and Wang et al. [22] also trained classifiers using aforementioned features [2]. While Bellogín et al. [4] trained a unique classifier for each target entity, Wang et al. [22] trained a general classifier for the whole target entities, achieving the best performance at TREC KBA 2013 [10]. The main reason why the general classifier outperformed per-entity classifiers is that the number of training instances was relatively small and thus insufficient if split across entities.

This study takes advantage of pseudo relevance feedback using non-relevant documents so as to remedy the problem of small training data and build a unique language model for each target entity. Most proposed approaches in vital filtering reported high recall and low

precision [10, 12], which means that there are a plenty of negative feedback documents that can be exploited in our proposed framework.

Despite the various approaches to vital filtering as summarized above, there is much room to improve for the performance in detecting web documents containing novel information. In fact, all the approaches in KBA 2013 [10] and 2014 [12] did not make a significant difference from a rather simple baseline [10]. It is presumably due to the fact that there were no features subsection representing documents containing novel information. For instance, while term-based cosine similarity between a document and the target entity would be an effective feature for identifying documents related to the entity [2], it has nothing to do with novelty.

In the present work, we propose statistical language modeling approach along the line of our previous work [13]. A statistical language model [20] is a probability distribution over sequences of words. There are some types of models, e.g., unigram language models, n-gram language models, and neural network language models. We use unigram language models in the present work for simplicity. We build two statistical language models describing documents containing novel information. One is built from a knowledge base article corresponding to the target entity. Then, a document is judged to contain novel information when the similarity between the language model and the document is lower. Note that lower similarity means a small overlap between the language model and the document, which potentially indicates the existence of novel information. Another language model is built from a collection of documents containing novel information. This model is intended to capture the common features (terms) to be used with novel information. Based on the model, a document is judged to contain novel information when the similarity between the language model and the document is higher.

In addition, those previous works mentioned above do not consider the processing time. In order to recommend web/social media documents to the editors of knowledge bases, it is important to achieve not only high accuracy in detecting novel information but also to process input data in realtime. To this end, our proposed framework is implemented on the distributed realtime computation system, Apache Storm, and is evaluated for its scalability.

3 Proposed Framework

The aim of this study is to detect web documents containing novel information related to a given entity. To this end, our proposed framework is developed on Storm, consisting of multiple filters using statistical language models. The next sub-sections first introduce Apache Storm and then our proposed framework and other details are presented.

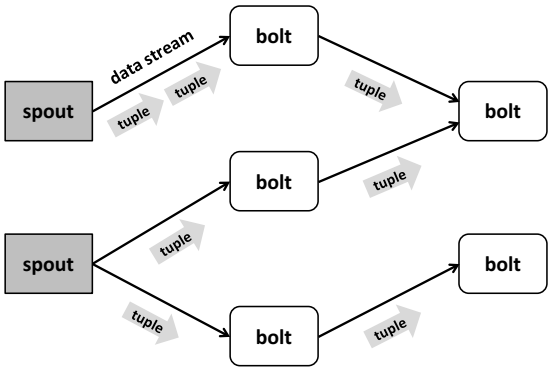


Figure 1: An illustration of Storm topology.

3.1 Apache Storm

Apache Storm is a distributed realtime computation system, processing unbounded streams of data. McCreadie et al. [17] proposed online event detection approach from high volume social streams using Storm. [21] describes the use of Storm at Twitter Inc.

To use Storm, one needs to define "topologies" illustrated in Figure 1. A topology is a graph of computation and each node in a topology has processing logic and edges between nodes indicate how data should be passed around between nodes.

There are two types of nodes, called "spouts" and "bolts". A spout is a source of streams (sequences of tuples) and a tuple is a unit of data processed in Storm. In case of our proposed framework, a spout would read document data from the provided corpus and emit them as a stream. A bolt receives any number of input streams, does any processing (e.g., running functions, filtering, and streaming aggregations), and may emit new streams. For our framework, bolts would determine whether inbound documents from the streams are relevant. Each node in a Storm topology executes in parallel and one can specify how much parallelism he/she wants for each node.

3.2 Overview of Framework

Figure 2 depicts the topology of our proposed framework, where input spout reads documents from a data source and sends them to surface form name (sfn) filter bolt, followed by relevant filter bolt, vital filter bolt, and so on. These bolts process the input stream of documents in parallel for a given target entity. Our frame-

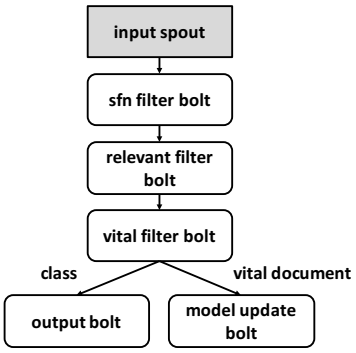


Figure 2: Topology of proposed vital filtering framework.

work consists of two-step filter: relevant filter detects relevant documents in input documents, vital filter detects vital documents in relevant documents. The following paragraphs provide brief descriptions of the each type of nodes of the topology.

First, **input spout** reads individual documents from the input stream and sends them to sfn filter bolts. Note that each document is preprocessed as described in Section 3.3.

Second, **sfn filter bolt** checks if each input document contains any surface form names of a given target entity. Only the documents containing the surface form name(s) are sent to the succeeding processes. There may be more than one surface form name for each entity. For instance, the surface form names of the entity "Barack Obama" are "Barack Obama", "Barack Hussein Obama", "Barack H Obama", and so on.

Third, **relevant filter bolt** judges whether an input document is relevant (i.e., vital or useful). If judged to be relevant, the document is sent to vital filter bolt. For this purpose, a negative language model built from a set of non-relevant documents for the target entity is used. More details are found in Section 3.4.

Fourth, **vital filter bolt** judges if a relevant document detected by our framework is vital or useful. The predicted class (i.e., vital or useful) are sent to output bolt. In addition, only the document judged to be vital is sent to **model update bolt** (see Section 3.6 for more details). The classification between vital or useful is based on two alternative language models; one is built from an article of knowledge base for the target entity, and another is built from a set of vital documents. The details are described in Section 3.5.

Lastly, **output bolt** collects and outputs the results of the preceding vital filter bolts. The following subsections give more details of the main components of the system as well as document preprocessing.

3.3 Document Preprocessing

Since our framework use statistical language models, all the input documents also need to be converted into statistical language models. For this reason, before processing documents with our framework, we preprocess all the input documents as follows:

- Uncapitalize words.
- Remove stop words and symbols.
- Apply the Porter stemmer [19] to deal with data sparseness.
- Build a document language model $p(w|d)$ for each document d with Dirichlet smoothing [25]:

$$p(w|d) = \frac{c(w,d) + \mu p(w|\mathcal{C})}{|d| + \mu} \quad (1)$$

where w denotes a word and Google-Ngram[3] (unigrams) is used to calculate the background language model $p(w|\mathcal{C})$. The value of μ is set to 2,000 following Zhai and Lafferty [25].

3.4 Relevant Filter Bolt

Relevant filter bolt judges whether an input document is relevant (i.e., vital or useful) or not. For this purpose, we use a negative language model (NLM) built from non-relevant documents for the target entity in question adopting the concept of MultiNeg [23], which has been shown effective for difficult queries where the search results are poor.

MultiNeg is a model to improve ad-hoc retrieval by negative relevance feedback, which takes advantage of pseudo feedback of non-relevant documents. More specifically, according to the similarity between the language models built from non-relevant documents and an input document, the relevance score of the document is adjusted. The non-relevant documents here mean those irrelevant to search intention within the initial search result. For example, consider the case where a user would like to search for information regarding Apple Inc. and used a query "apple". The search results would contain documents regarding apples (fruit), which are non-relevant in this case.

MultiNeg builds an NLM, $\Theta = \{\theta_1, ..., \theta_f\}$, for each of such irrelevant documents $L = \{l_1, ..., l_f\}$ using the standard EM algorithm [7]. In MultiNeg, the relevance score of the document, $S(q,d)$, is defined by the KL-divergence retrieval model [14], computed based on the

[3] http://googleresearch.blogspot.jp/2006/08/all-our-n-gram-are-belong-to-you.html

negative KL-divergence between query model θ_q and document model θ_d, i.e.,

$$S(q,d) = -D(\theta_q||\theta_d) = -\sum_{w \in V} p(w|\theta_q) \log \frac{p(w|\theta_q)}{p(w|\theta_d)} \tag{2}$$

where V is a vocabulary of the language models. Then, adjusted relevance score is defined as follows:

$$S(q,d) - S(NLM,d) \tag{3}$$

where $S(NLM,d)$ is penalty term indicating the similarity between document d and non-relevant documents as defined in Eq. (4).

$$S(NLM,d) = \quad -\min(\bigcup_{i=1}^{f}\{D(\theta_i||\theta_d)\}) \tag{4}$$

This study uses Eq. (4) to filter out irrelevant documents. Specifically, our framework judges document d as relevant when Eq. (5) is satisfied:

$$S(NLM,d) < t_r \tag{5}$$

where t_r is predefined threshold.

3.5 Vital Filter Bolt

Vital filter bolt judges if an input document is vital or useful using two alternative language models, that is, Knowledge base Article Language Model (KALM) and Vital Language Model (VLM). KALM is a unigram language model, built from an article of knowledge base for the target entity. In addition, known vital documents (i.e., training data) for the target entity are used in building KALM because the information in those vital documents should be included in the corresponding article. VLM is also a unigram language model, built from a set of known vital documents.

To build VLM, we first identify the terms characterizing vital documents based on chi-square statistic [16] using a vital document set vs. a useful document set in the training data. Chi-square statistic is often used for identifying good features (i.e., feature selection) in classification. Table 1 shows the contingency table, where V_+ (V_-) denotes the number of documents when a term w in question appeared (did not appear) in the vital document set. Similarity, U_+ (U_-) are the number of documents when w appeared (did not appear) in the useful document set. Chi-square statistic χ^2 is calculated for each term w for each entity as in Eq. (6).

$$\chi^2 = \sum_{i \in \{+,-\}} \frac{(V_i - E_i^V)^2}{E_i^V} + \sum_{i \in \{+,-\}} \frac{(U_i - E_i^U)^2}{E_i^U} \tag{6}$$

Table 1: Notation for computing chi-square statistic.

	Vital	Useful	Sum
Appear	V_+	U_+	N_+
Not appear	V_-	U_-	N_-
Sum	V	U	N

where $E_i^V = N_i \cdot V/N$ and $E_i^U = N_i \cdot U/N$. The result of chi-square test shows whether or not the frequency of w is significantly different between vital document set and useful document set. If different, w can be seen as a good feature to distinguish between vital and useful documents. Among the terms with high chi-square statistic, those satisfying Eq. (7) are excluded.

$$V_+ < V\frac{N_+}{N} \cap \chi^2 < 3.84 \tag{7}$$

where 3.84 is the chi-square statistic at the significance level of 5% for one-degree-of-freedom. The remaining terms are used as the vocabulary of VLM, and VLM is built as the same processes as the input document (see Section 3.3). Note that chi-square statistic cannot be computed when there is no useful document. In such cases, all the terms in vital documents are used as the vocabulary.

In filtering, the similarity between the language models and input documents are computed as the negative KL-divergence as in Eq. (2), from which the document is judged to be vital or not (i.e., useful). For KALM, the document is judged as vital if the similarity is *lower* than a predefined threshold t_{vk}. The rationale behind is that a document more different from an article of knowledge base would contain more, and possibly relevant, information not described in the article. For VLM, conversely, the document is judged as vital if the similarity is *greater* than a predefined threshold t_{vv}. The assumption here is that the vocabularies often found in vital documents, and the language model built from it, would capture some patterns of words characteristic to vital documents. Hence, a document similar to it would be also vital.

3.6 Model Update Bolt

Model update bolt receives a document which is judged as vital in the preceding vital filter bolt and updates KALM or VLM. Specifically, the number of occurrences, denoted as $c'(w,d)$, of each term w in the received document d' is added to the current statistics, $c(w,d)$ in Equation (1), of KALM/VLM. Note that terms with low chi-square statistic are not used to update VLM. The updated model is sent back to vital filter bolt and will

be used afterwards. This process simulates the editing of knowledge base articles in the light of new information related to the target entities.

4 EVALUATION

This section empirically evaluates our proposed framework by conducting a set of experiments. The experiments were designed to examine the detection performance and processing time of the framework. The next sub-sections first describe the experimental settings and then provide and discuss the results of the experiments.

4.1 Experimental Settings

We follow the evaluation methodology adopted at the TREC 2014 KBA vital filtering task [12]. The KBA track provided its participants with a large corpus, called the TREC KBA Stream Corpus 2014[4]. This corpus covers the time period from October 2011 to April 2013, containing 20,494,260 documents, including blogs, forum posts, and web pages. Each document in the corpus is associated with a time-stamp corresponding to its date of publication. Documents within the predefined time range is available as training data. The number of target entities is 67. The macro-averaged F_1 (harmonic mean between precision and recall) was used as the evaluation metric.

$$F_1 = \frac{2 \cdot P_{ave} \cdot R_{ave}}{P_{ave} + R_{ave}} \tag{8}$$

$$P_{ave} = \frac{1}{|E|} \sum_{e \in E} P(e) \tag{9}$$

$$R_{ave} = \frac{1}{|E|} \sum_{e \in E} R(e) \tag{10}$$

where $P(e)$ and $R(e)$ are precision and recall for entity e, respectively, and E denotes the set of 67 target entities. Note that low recall and high precision leads to less documents to manually inspect but it may miss important documents. On the other hand, high recall and low precision leads to more documents to review, which may not be feasible if the number of editors is limited. Since which scenario is preferable depends on the number of editors for each entity, the evaluation in this paper focuses on F_1.

We used canonical names as the surface form names of target entities. The canonical names were provided along with the Stream Corpus by the TREC KBA organizers. In addition, for those entities which have their Wikipedia articles, redirect[5] information extracted from the Wikipedia dump on 1/4/2012[6] were also utilized.

To estimate NLM, θ_i, our system used non-relevant documents in the training data in the Stream Corpus. We considered documents which contain surface form name(s) of a target entity but do not have a "vital" or "useful" label as non-relevant documents. Note that if the number of documents containing a surface form name is too large (>100k for our experiments), the name is unlikely informative and thus was not utilized. Also, if a target entity did not have non-relevant documents, relevant filter bolt was disabled for the entity. The threshold t_r was tuned to the smallest value among the thresholds based on which vital or useful documents in the training data would be judged to be relevant.

To build KALM, we used the articles in the Wikipedia dump on 1/4/2012 for entities which have Wikipedia articles. The same preprocessing described in Section 3.3 was applied to the extracted articles. Vital and useful documents in the training data were used as known vital/useful document set to build VLM. The threshold t_{vk} and t_{vv} were set for each entity using the vital and useful documents in the training data such that F_1 is maximized based on their similarity scores with the language model.

It should be mentioned that the results to be shown in the next sections are not exactly the same as the results reported in the TREC KBA official report [12] although we used the data from the KBA track. The difference is due to how unlabeled data were treated. The KBA data are incomplete, meaning that not all documents are labeled and the KBA official results ignored them in computing precision, where simply judging all documents as positives results in the perfect precision. A more realistic scenario would be treating unlabeled documents as negatives since almost all documents are negatives. By adopting this scenario, unlabeled documents were treated as negatives in the following experiments.

4.2 Results and Discussion

This section first discusses the effect of the relevant filter bolt, then the effect of the vital filter bolt with/without model updates. In addition, processing time is examined to evaluate the proposed framework for realtime stream data processing.

Table 2 shows the result of relevant documents detection, where "NLM" is the negative language model-based filter described in Section 3.4 and "Exact Match" simply treated the documents which went through sfn filter bolt as vital. That is, the documents containing surface form name(s) of the target entities were unconditionally judged as vital. Exact Match was used as the

[4] http://s3.amazonaws.com/aws-publicdatasets/trec/kba/index.html
[5] http://en.wikipedia.org/wiki/Wikipedia:Redirect
[6] http://s3.amazonaws.com/aws-publicdatasets/trec/kba/enwiki-20120104/index.html

Table 2: Performance of relevant documents detection where bold figures indicate better results.

Setting	Precision	Recall	F_1
Exact Match	0.285	**0.995**	0.443
NLM	**0.313**	0.978	**0.474**

Table 3: Performance of vital documents detection where bold figures indicate best results.

Setting	Precision	Recall	F_1
Exact Match	0.106	**0.993**	0.192
KALM (w/o update)	0.117	0.884	0.206
KALM	0.120	0.876	0.211
VLM (w/o update)	**0.153**	0.428	0.225
VLM	0.139	0.641	**0.228**

baseline in the KBA vital filtering task [10, 12]. Although Exact Match is simple, it was reported to be quite difficult to beat the simple baseline. Notice that the difference of NLM and Exact Match directly indicates the effect of our relevant filter.

The additional filter implemented by NLM decreased recall by 1.7% ($= 1 - 0.978/0.995$), whereas precision increased by 9.8% ($= 0.313/0.285 - 1$). As a result, F_1 improved from 0.443 to 0.474 (+7.0%). The improvement of F_1 was found statistically significant at the 1% significance level by pair-wise t-test, which demonstrates the effectiveness of our filter based on negative language models built from irrelevant documents.

Then, Table 3 shows the performance of vital documents detection, where "KALM" used the language model based on knowledge base article and "VLM" used the language model based on vital documents in vital filter bolts (see Section 3.5). Note that "w/o updates" indicates that the system did not use the model update bolt, where KALM or VLM was static at runtime.

As Table 3 shows, both KALM and VLM improved on the baseline, suggesting the effectiveness of our language model-based filters. The model update also brought small but consistent positive effects for KALM and VLM in F_1. To be more precise, KALM has higher precision and lower recall, whereas VLM shows lower precision and higher recall after model update. The opposite behaviors of KALM and VLM are due to the fact that the similarity between KALM/VLM and a document tend to be higher because of the new terms added to KALM/VLM by model update. Consequently, the number of documents judged as vital decreases for KALM and increases for VLM. Remember that, for KALM, a document is judged as vital if the similarity is lower than a threshold, and the opposite is true for VLM.

VLM with model update yielded the best result on average among the different system settings. However, significant difference was found only between Exact Match and KALM ($p < 0.05$), that is, there was no significant difference between Exact Match and VLM.

On investigation, it was found that the result was due to the larger variance of the performance of VLM than that of KALM as contrasted in Figure 3 and Figure 4. The bar graphs show the difference of F_1 between KALM/VLM and Exact Match for each target entity. For KALM, the performance improved on the baseline (Exact Match) for the majority of the entities, even though around a half are marginal. On the other hand, while VLM's improvement is more noticeable than that of KALM, it also showed strong negative effects for some entities (the most noticeable two in Figure 4 are "Lizette Graden[7]" and "Corisa Bell[8]"). We will discuss this issue in the next sub-section to circumvent the problem.

4.3 Hybrid Model

As mentioned above, VLM's performance is greater than that of KALM on average but VLM is suffered from a few target entities on which VLM has severe negative effects. We looked into the problem and found that it was caused by the fact that the size of training data is largely different from entities to entities.

Specifically, there exist eight target entities for which there is no useful documents. Remember that vital documents and useful documents are used in computing chi-square statistic in order to determine the vocabulary of VLM (see Section 3.5). When there is no useful documents, the statistic cannot be computed. Therefore the previous experiment used as the vocabulary all the terms occurring in vital documents for those target entities.

To deal with the problem, we constructed a simple but effective hybrid model incorporating KALM into VLM. To be precise, the hybrid model primarily used VLM as the vital filter and switched to KALM only when the target entity in question had no useful document. Table 4 summarizes the result.

As a result, the hybrid vital filter achieved the best F_1 score of 0.245. This time, the improvement was statistically significant at the 1% significance level when compared with Exact Match.

[7]Lizette Graden is a chief curator at Nordic Heritage Museum.
[8]Corisa Bell is a councillor in Maple Ridge, Canada.

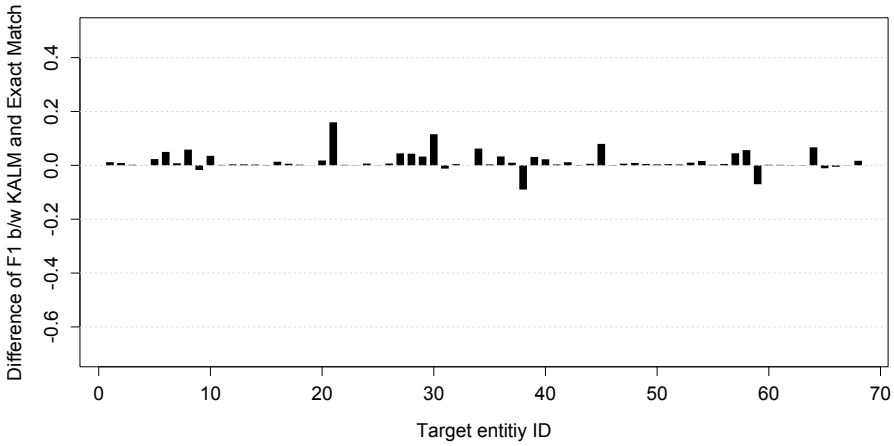


Figure 3: Per-topic difference of F_1 measures between KALM and Exact Match for each entity.

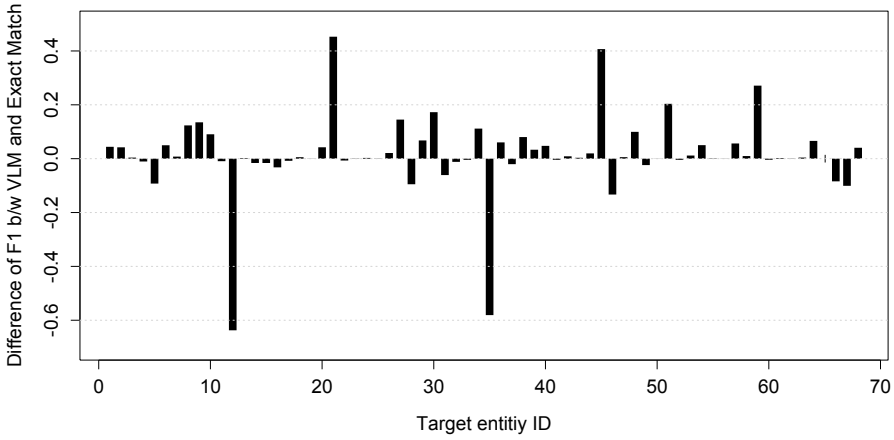


Figure 4: Per-topic difference of F_1 measures between VLM and Exact Match for each entity.

Table 4: Performance comparison of the hybrid model and other models where bold figures indicate the best results.

Setting	Precision	Recall	F_1
Exact Match	0.106	**0.993**	0.192
KALM	0.120	0.876	0.211
VLM	0.139	0.641	0.228
Hybrid	**0.147**	0.724	**0.245**

4.4 Realtime Processing

Lastly, we evaluated the performance of our proposed filtering framework from the viewpoint of processing time. As mentioned, our goal was to process unbound data streams flooding on the Web to find vital documents for a given target entity, where it is critical that it is able to process the data streams in realtime. To this end, Figure 5 shows the processing time required to process a sample (around 16GB) of the TREC KBA Stream Corpus 2014 with a different number of relevant filter bolts, where relative speed-up is also shown for information. It should be mentioned that we ignored vital filter bolts in this experiment because it has little effect on the processing time.

We can observe that when the number of bolts was increased, the processing time became smaller. Then, it became flat at around six minutes (360 seconds), where the number of relevant filter bolts was around five to six. As for relative speed-up, the processing speed almost linearly increased with the number of relevant filter bolts until the number of bolts reached five, above which there appears to be a ceiling effect. However, this is due to *not* the limitation of our proposed framework but the limitation of input data sources. We measured the pro-

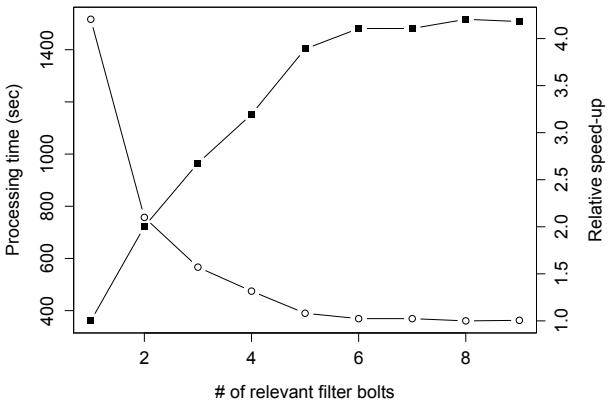


Figure 5: Processing time and relative speed-up with an increasing number of relevant filter bolts.

cessing time of only input spout for reading and sending all the documents in the input data, which took about six minutes. In other words, the processing time in Figure 5 can never be lower than six minutes. This fact and the nearly linear scalability mentioned above suggest that our framework would scale out to an even greater amount of input data by simply adding more bolts.

5 Summary and Conclusions

In this paper, we proposed a framework to detect vital Web documents containing novel information in massive document streams. Our framework employed a language model-based approach and used irrelevant documents for identifying relevant (vital or useful) documents and articles of knowledge base and known vital documents for further identifying vital documents. The language models, KALM and VLM were updated every time a vital document was identified. The framework was implemented as a distributed realtime processing system, improving on the strong baseline using surface form names of the target entities. Moreover, when the language models, KALM and VLM, were jointly used, the performance measured in the macro-averaged F_1 for detecting vital documents outperformed the individual models. Furthermore, we demonstrated that our proposed framework would be able to process massive document streams in realtime by increasing the number of bolts.

There are several directions to improve our framework. One is to incorporate time-aware features, which have been considered effective to detect vital documents [2, 5]. We are planning to combine time-aware features with textual features used in our framework.

Acknowledgments

This work is partially supported by JSPS KAKENHI Grant Number 25330363 and MEXT, Japan.

References

[1] R. Abbes, K. Pinel-Sauvagnat, N. Hernandez, and M. Boughanem, "IRIT at TREC Knowledge Base Acceleration 2013: Cumulative Citation Recommendation Task," in *Proceedings of the TExt Retrieval Conference (TREC)*, 2013.

[2] K. Balog, H. Ramampiaro, N. Takhirov, and K. Nørvåg, "Multi-step classification approaches to cumulative citation recommendation," in *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*, 2013, pp. 121–128.

[3] K. Balog, P. Serdyukov, and A. P. d. Vries, "Overview of the TREC 2011 Entity Track," in *Proceedings of the Text REtrieval Conference (TREC)*, 2011.

[4] A. Bellogín, G. G. Gebremeskel, J. He, J. Lin, A. Said, T. Samar, A. P. de Vries, and J. B. Vuurens, "CWI and TU Delft at TREC 2013: Contextual suggestion, federated web search, KBA, and web tracks," in *Proceedings of the TExt Retrieval Conference (TREC)*, 2013.

[5] L. Bonnefoy, V. Bouvier, and P. Bellot, "A weakly-supervised detection of entity central documents in a stream," in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, 2013, pp. 769–772.

[6] H. T. Dang, D. Kelly, and J. J. Lin, "Overview of the TREC 2007 Question Answering Track," in *Proceedings of the Text REtrieval Conference (TREC)*, 2007.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the royal statistical society, series B*, vol. 39, no. 1, pp. 1–38, 1977.

[8] L. Dietz and J. Dalton, "UMass at TREC 2013 Knowledge Base Acceleration Track: Bi-directional Entity Linking and Time-aware Evaluation," in *Proceedings of the TExt Retrieval Conference (TREC)*, 2013.

[9] J. L. Elsas, J. Arguello, J. Callan, and J. G. Carbonell, "Retrieval and feedback models for blog feed search," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 347–354.

[10] J. R. Frank, S. J. Bauer, M. Kleiman-Weiner, D. A. Roberts, N. Tripuraneni, C. Zhang, C. Re, E. Voorhees, and I. Soboroff, "Evaluating Stream Filtering for Entity Profile Updates for TREC 2013 (KBA Track Overview)," in *Proceedings of the Text REtrieval Conference (TREC)*, 2013.

[11] J. R. Frank, M. Kleiman-Weiner, D. A. Roberts, F. Niu, C. Zhang, C. Ré, and I. Soboroff, "Building an entity-centric stream filtering test collection for TREC 2012," in *Proceedings of the Text REtrieval Conference (TREC)*, 2012.

[12] J. R. Frank, M. Kleiman-Weiner, D. A. Roberts, E. Voorhees, and I. Soboroff, "Evaluating stream filtering for entity profile updates in TREC 2012, 2013, and 2014," in *Proceedings of the TExt Retrieval Conference (TREC)*, 2014.

[13] S. Kawahara, K. Seki, and K. Uehara, "Detecting Vital Documents Using Negative Relevance Feedback in Distributed Realtime Computation Framework," in *Proceedings of the 2015 Conference of the Pacific Association for Computational Linguistics*, 2015, pp. 101–108.

[14] J. Lafferty and C. Zhai, "Document language models, query models, and risk minimization for information retrieval," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 111–119.

[15] X. Liu, J. Darko, and H. Fang, "A Related Entity based Approach for Knowledge Base Acceleration," in *Proceedings of the TExt Retrieval Conference (TREC)*, 2013.

[16] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press, 2008.

[17] R. McCreadie, C. Macdonald, I. Ounis, M. Osborne, and S. Petrovic, "Scalable distributed event detection for twitter," in *2013 IEEE International Conference on Big Data*, 2013, pp. 543–549.

[18] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, 2007, pp. 233–242.

[19] M. F. Porter, "An algorithm for suffix stripping," *Program: electronic library and information systems*, vol. 14, no. 3, pp. 130–137, 1980.

[20] F. Song and W. B. Croft, "A general language model for information retrieval," in *Proceedings of the eighth international conference on Information and knowledge management*, 1999, pp. 316–321.

[21] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham *et al.*, "Storm@ twitter," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 147–156.

[22] J. Wang, D. Song, C.-Y. Lin, and L. Liao, "BIT and MSRA at TREC KBA CCR Track 2013," in *Proceedings of the TExt Retrieval Conference (TREC)*, 2013.

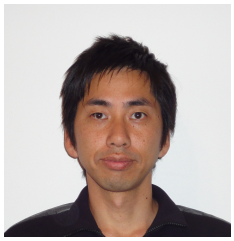
[23] X. Wang, H. Fang, and C. Zhai, "A study of methods for negative relevance feedback," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 219–226.

[24] Y. Xu, G. J. Jones, and B. Wang, "Query dependent pseudo-relevance feedback based on wikipedia," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 2009, pp. 59–66.

[25] C. Zhai and J. Lafferty, "A study of smoothing methods for language models applied to Ad Hoc information retrieval," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 334–342.

AUTHOR BIOGRAPHIES

Shun Kawahara was born in 1991. He received his B.E. in engineering from Kobe University. His research interests include machine learning and natural language processing. He is currently an M.E. student in the Graduate School of System Informatics, Kobe University.



Dr. Kazuhiro Seki received his Ph.D. in information science from Indiana University, Bloomington. His research interests are in the areas of natural language processing, information retrieval, and data mining. He is currently an associate professor in Faculty of Intelligence and Informatics at Konan University.

Dr. Kuniaki Uehara received his B.E., M.E. and D.E. degrees in information and computer sciences from Osaka University, Japan. He was an assistant professor in the Institute of Scientic and Industrial Research at Osaka University and was a visiting assistant professor at Oregon State University. Currently, he is a professor in the Graduate School of System Informatics at Kobe University, Japan. His is conducting research in the areas of machine learning, data mining, and multimedia processing. He is a member of the Japanese Society for Artificial Intelligence, Japan Society for Software Science and Technology, Information Processing Society of Japan, and AAAI.